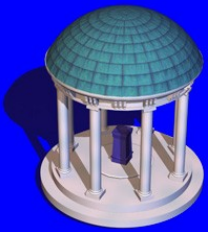


The Design of Design

Fred Brooks

**University of North Carolina
at Chapel Hill**

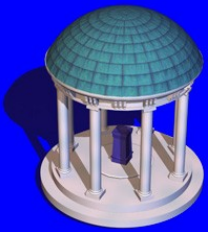
brooks@cs.unc.edu



Design

**“To form a plan or scheme of,
to arrange or conceive in the
mind...
for later execution.”**

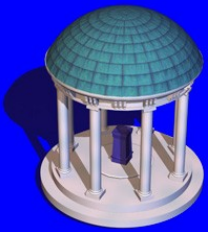
Oxford English Dictionary



W.A. Mozart

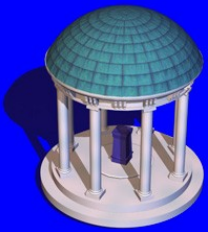
"Everything has been
composed,
just not yet written down."

Letter to Leopold Mozart,
1780



Why Study the Design Process?

- Can I design better by looking at design process?
- Can we better teach others to design?
- Can we better organize and manage design?
- Let's limit ourselves to the design of systems

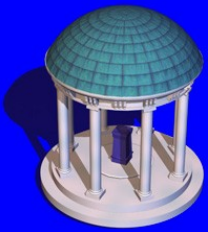


Sir Francis Bacon's Reason

- New ideas would come about “by a connexion and transferring of the observations of one Arte, to the uses of another, when the experience of several misteries shall fall under consideration of one mans minde.”

The Two Books of the Proficience and Advancement of Learning, Book 2, p 10, 1605

- **Design process studied in architecture, mechanical engineering, and industrial design.**
- **What can we learn from them?**

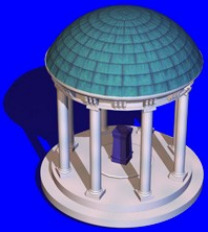


21st Century Design Issues

I. Models of the design process

II. Collaborative teams and solo/chief designers

III. How to get *great* designs



I. How Engineers Think of Design

- Goal
- Desiderata
- (Non-linear) utility function
- Constraints, especially budget (not necessarily linear)
- Design tree of decisions

UNTIL (design is "good enough") or (time has run out)

DO another design (to improve utility function)

UNTIL design is complete

WHILE design remains feasible,
make another design decision

END WHILE

Backtrack up design tree

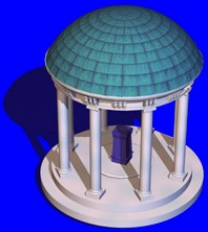
Explore a path not searched before

END UNTIL

END DO

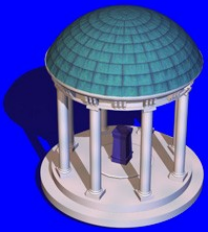
Take best design

END UNTIL



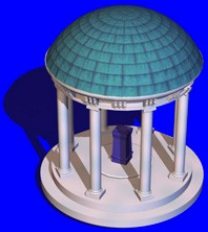
What's Wrong with This Model?—1

- **We don't really know the goal at first –**
 - The hardest part of design is deciding *what* to design.
 - Often the most important function of the designer is helping the client decide what he wants.



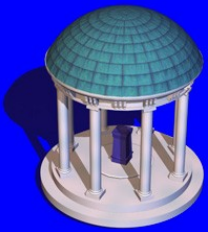
What's Wrong with This Model?—1

- We don't really know the goal at first -
- Here is where experts go wrong:
 - Miss fresh vision - e.g., minicomputer, microcomputer
 - Vision not high enough - e.g., OS/360 JCL



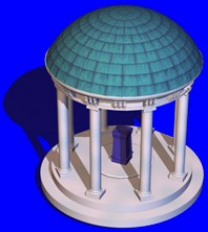
OS/360 JCL - the Worst Language

- Done under my management
- One job language for all programming languages
- Like Assembler language, rather than PL/I, etc.
- But not exactly like Assembler
- Card-column dependent
- Too few verbs
- Declarations do verbish things, in the guise of pa
- Awkward branching
- No clean iteration
- No clean subroutine call
- Basic problem was pedestrian vision
 - We did not see it as a schedule-time program at all, but as a "few control cards"
 - It was not *designed*, it just grew as needs ap



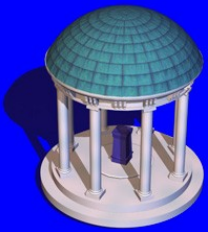
What's Wrong with This Model?—2

- The desiderata and their weightings keep
 - Donald Schön - "as one wrestles with the problem"
 - As one in fact *makes* the trade-offs, the weightings change
 - Sometimes one hits new opportunities.



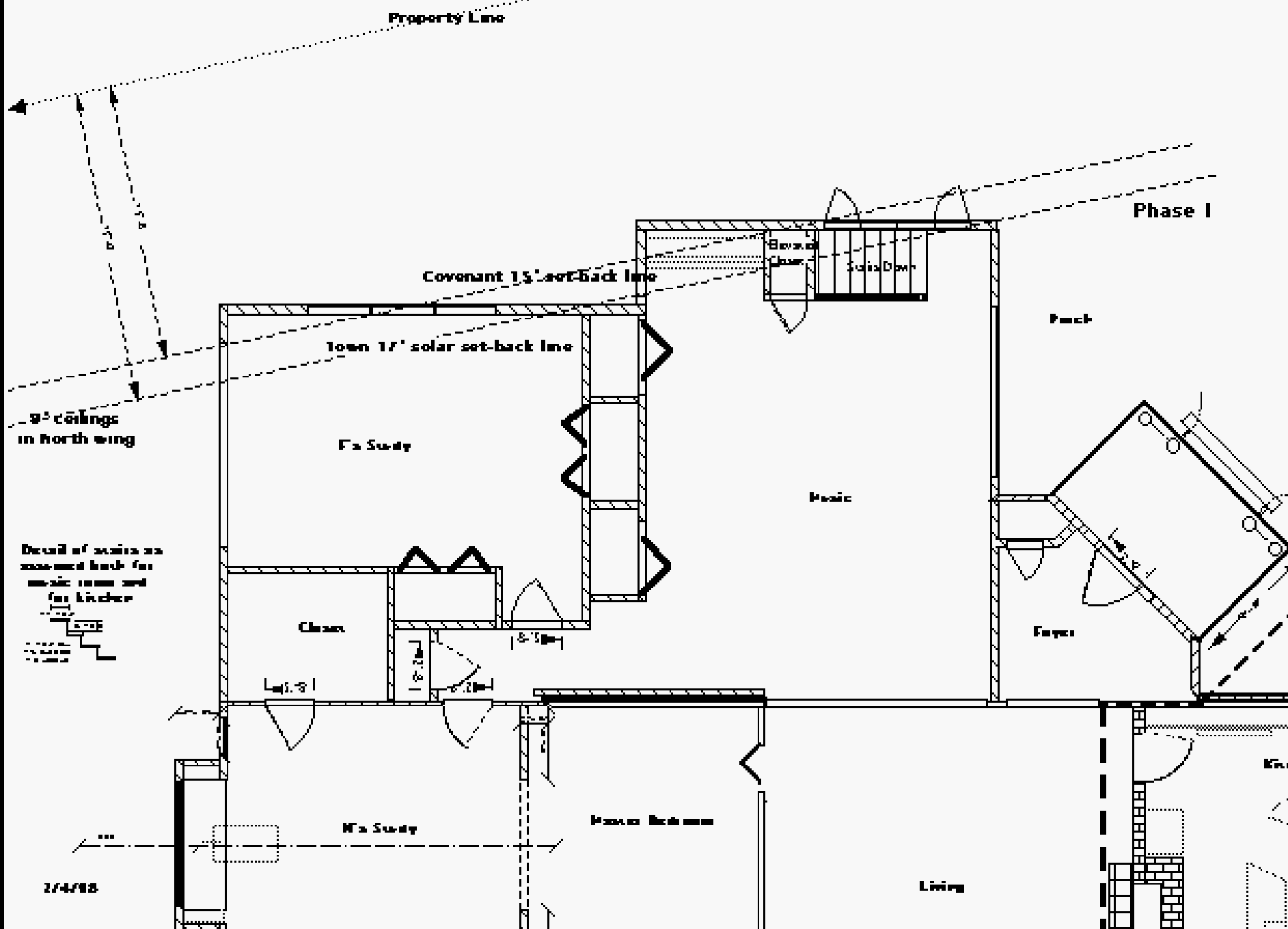
What's Wrong with This Model?—2

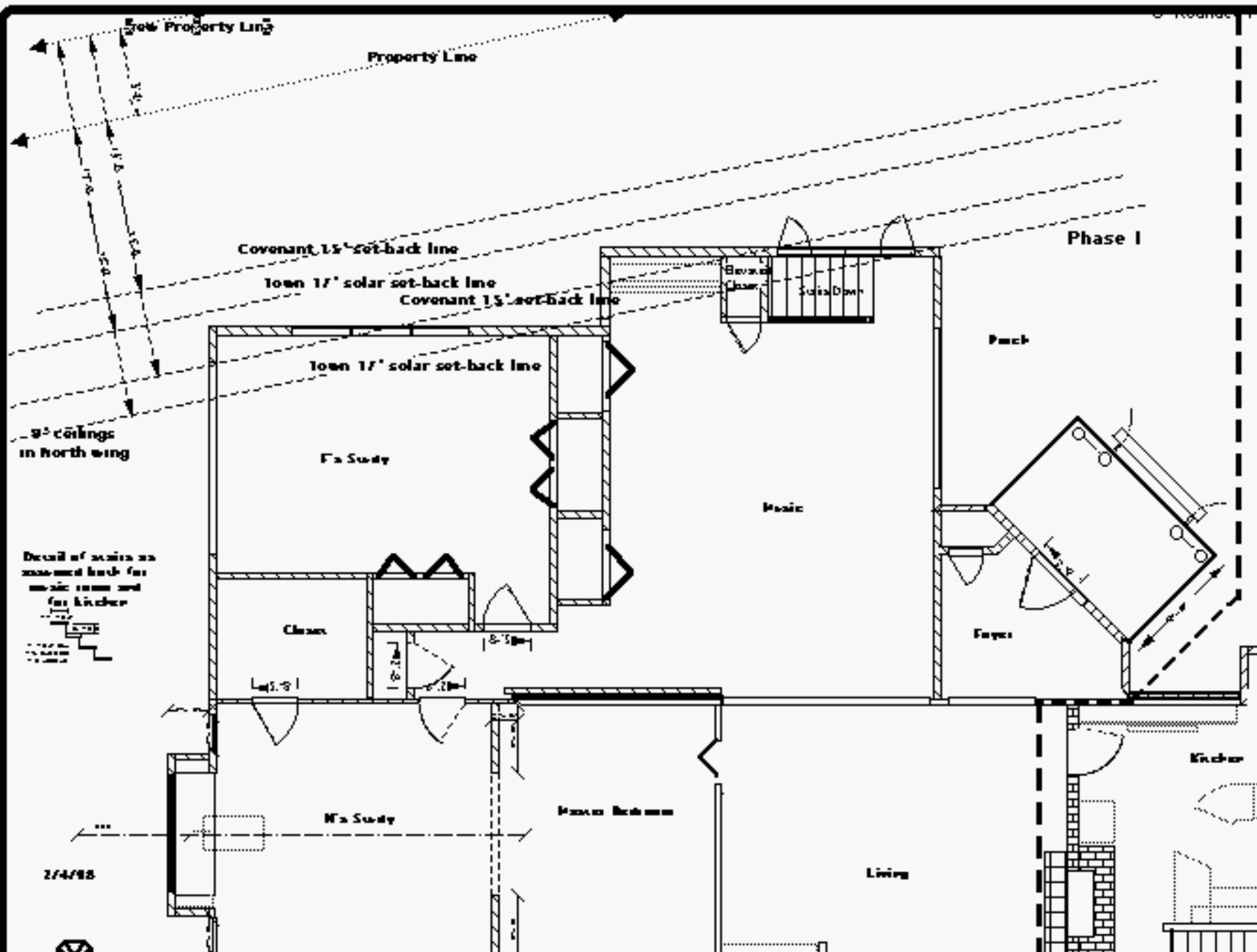
- **The desiderata and their weightings keep**
- **We usually don't know the design tree**



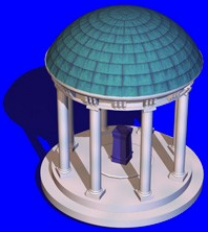
What's Wrong with This Model?—2

- **The desiderata and their weightings keep**
- **We usually don't know the design tree.**
- **The constraints keep changing.**
 - **Often by the ever-changing world.**
 - **Sometimes by total systems thinking, outside the box!**



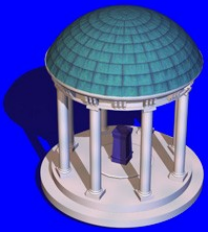


2/4/88



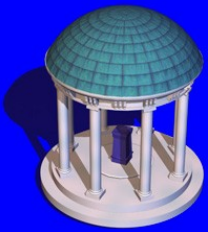
Design Models

- The rational model is wrong—doesn't describe what really goes on
 - But still the “consensus model” in engineering literature.
 - See for example, G.Pahl and W. Beitz, 1984 *Engineering Design: A Systematic Approach*



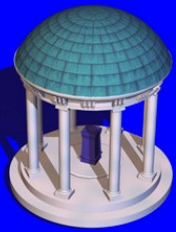
Design Models

- The rational model is wrong—
doesn't describe what really goes on
- Most expert designers don't work
that way



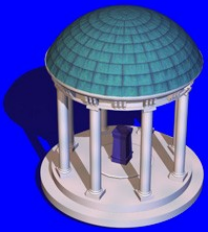
Design Models

- The rational model is wrong—
doesn't describe what really goes on
- Most expert designers don't work that way
- It can give bizarre results
 - LHX helicopter functional specs



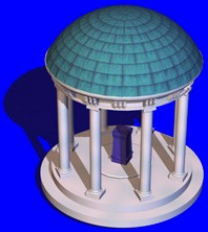
Design Models

- The rational model is wrong—
doesn't describe what really goes on
- Most expert designers don't work that way
- It can give bizarre results
- We have such a model in software engineering



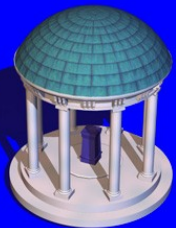
The Waterfall Model is Dead Wrong

- Not how we buy a new building, or a new airplane
 - We pay for a *design phase*, approve a design, and contract for its implementation, or
 - A builder pays for a design phase, sells implementations



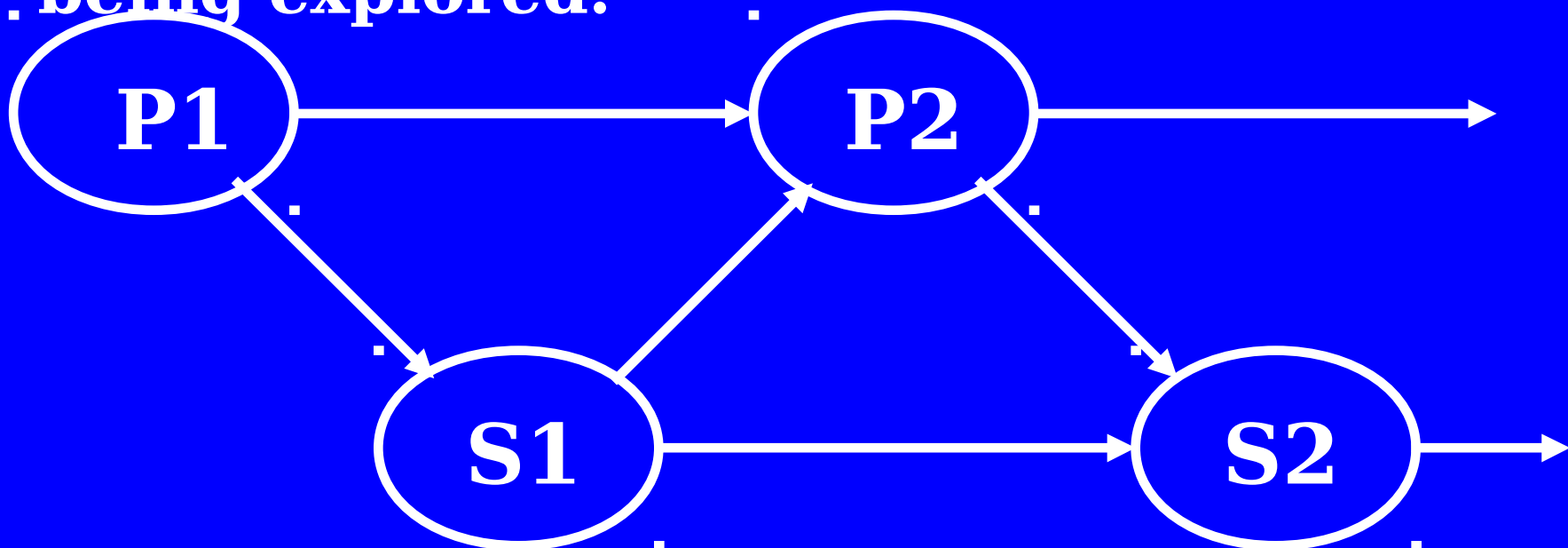
The Waterfall Model is Dead Wrong

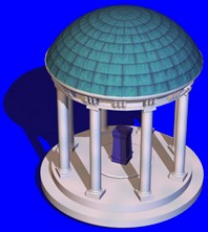
- **Not how we buy a new building, or a new airplane**
- **Based on spurious assumption that function and performance are what matters about software.**
 - **A naive notion from early days**
 - **Reliability, changeability, structure, testability**



A Better Model —Co-evolution

- Model due to Maher, Cross
- The effective problem space evolves as the solution space evolves by being explored.





Evolutionary Software Development

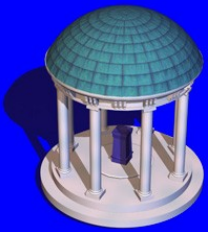
1. Build a minimal working system.

2. Try it with real users.

3. Revise.

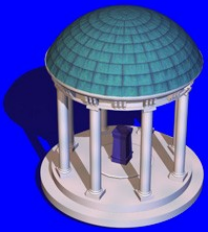
4. Add function in small increments.

- Robust under changing desiderata and conditions
- Early testing exposes our inevitable mistakes



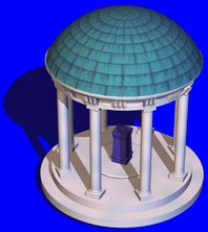
II. Solo Design and Collaboration

- The design *team* is the 20th-century novelty.
- *Conceptual integrity* is the problem with this —
hard!
- Design as “interdisciplinary negotiation”? NO!
- Mills’ chief-programmer concept —



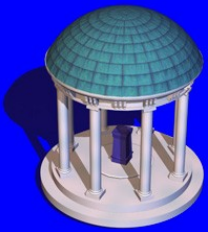
A System Architect

- A *system architect*,
for designs beyond one chief designer
- The architect:
agent, approver, advocate for the *user*
- Detailed: Chapters 3-6 in
The Mythical Man-Month



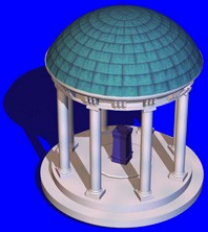
The Cathedral and the Bazaar

- Raymond's brilliant essays on Linux's process
- The bazaar is an evolutionary model
 - No committee design — each piece has conceptual integrity
 - Emphasizes early and *large-scale* testing
 - Marshalls many minds for fixing, not just testing
 - The market votes by adoption among alternatives
- Based on a gift \longleftrightarrow prestige culture



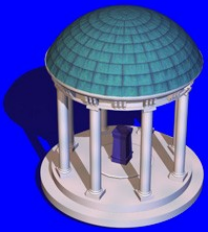
The Cathedral and the Bazaar

- Raymond's brilliant essays on Linux's process
- The bazaar is an evolutionary model
- Based on a gift \longleftrightarrow prestige culture
- Among people who are fed anyway
- Works when the builders are the clients
 - Know requirements from personal experience
- Is this how to do an air traffic control system?



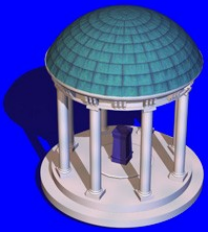
Collaboration and Telecollaboration

- **Collaboration is politically correct
and fashionable.**



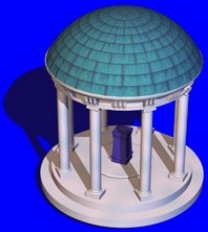
Collaboration and Telecollaboration

- **Collaboration is politically correct and fashionable.**
- **Telecollaboration is even more so.**



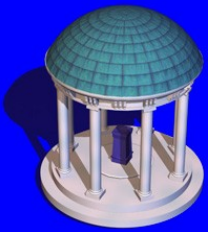
Collaboration and Telecollaboration

- **Collaboration is politically correct and fashionable.**
- **Telecollaboration is even more so.**
- **Much telecollaboration R & D is technology-pushed, not application-pulled.**



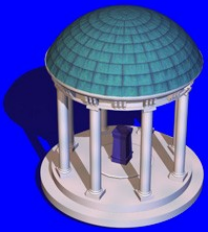
Collaboration and Telecollaboration

- **Collaboration is politically correct and fashionable.**
- **Telecollaboration is even more so.**
- **Much telecollaboration R & D is technology-pushed, not application-pulled.**
- **We need far more understanding of collaboration.**



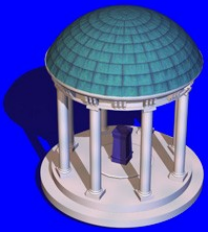
Design Collaboration

- Real design is always more complex than we imagine.
- E.g., fixtures for parts, tooling limitations, assembly



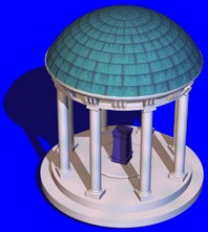
Design Collaboration

- Real design is always more complex than we imagine.
- E.g., fixtures for parts, tooling limitations, assembly
- Real design is hard to change —
Design Change Control



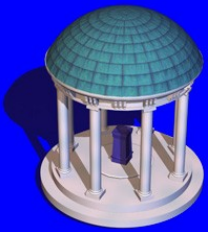
Design Collaboration

- Real design is always more complex than we imagine.
- Design change control
- The cleaner the interfaces, the fewer errors.
 - Errors and rework are the big cost components.
 - Hence, *constrained* collaboration is most productive.



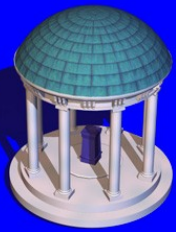
Design Collaboration

- Real design is always more complex than we imagine.
- Design change control
- The cleaner the interfaces, the fewer errors.
- Collaboration is no substitute for “the dreariness of labor and the loneliness of thought.”



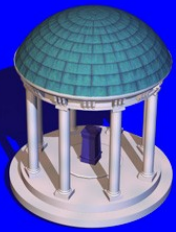
When Does Collaboration Help?

- **Determining needs from users**
 - **More minds —> more diverse questions**



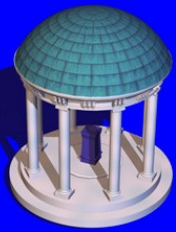
When Does Collaboration Help?

- Determining needs from users
- Conceptual exploration — radical alternatives



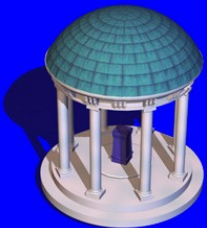
When Does Collaboration Help?

- **Determining needs from users**
- **Conceptual exploration — radical alternatives**
- ***Not* conceptual design nor detailed design**
 - **Observe the great works of the human mind**



When Does Collaboration Help?

- **Determining needs from users**
- **Conceptual exploration — radical alternatives**
- ***Not* conceptual design nor detailed design**
- **Design reviews**
 - **Especially with different expertise**
 - **Need and exploit richer graphical representations**



III. Great Designs

Fan Clubs

No Fan Clubs

Fortran

COBOL

VM/360

OS/360

Unix, Linux

Microsoft NT

Pascal

Algol

C

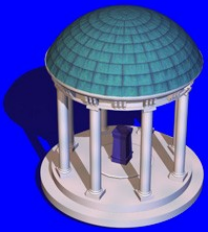
PL/I

Macintosh

PC

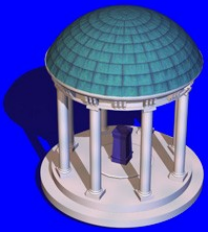
APL

Ada



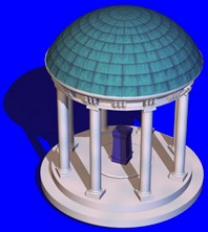
Product Processes

- **Within- vs. outside product-process;**
What are product procedures for?



Product Processes

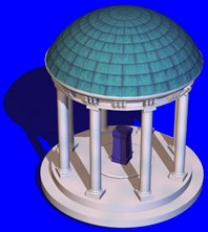
- **Within-product-process vs. outside-product-process**
What are product procedures for?
- **How to do great design *within* a product process?**
- **How to make a product process that encourages, rather than inhibits, great designs?**



Great Designs Come

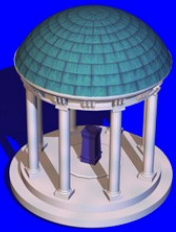
From Great Designers

- How does one do great designs *within* a process?
- How to make a product process that encourages rather than inhibits, great designs?
- Where elitism is proper
- Entrust design to a chief designer



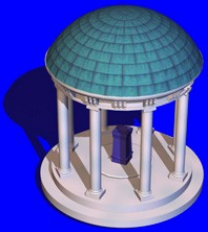
Where Do Great Designers Come From?

- We have to grow them *deliberately*.
- Recruit for design brilliance, not talk skills
- Make the dual ladder real and honorable
- Career planning and mentoring, as for non-designers
- Planned experiences, studies, and rotations



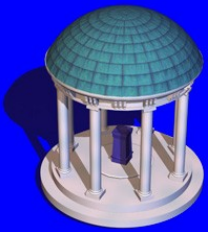
Where Do Great Designers Come From?

- We have to grow them *deliberately*.
- We have to manage them *imaginatively*
 - The John Cocke story



Where Do Great Designers Come From?

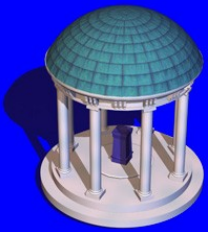
- We have to grow them *deliberately*.
- We have to manage them *imaginatively*.
- We have to protect them *fiercely*.
 - From managers
 - From managing



Growing Yourself as a Designer

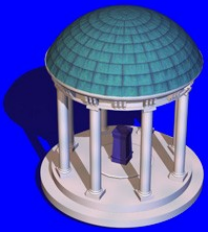
- Design lots of things, and keep a notebook.
 - Da Vinci's *Notebooks*
- Reflect in writing on your design experiences.
- Study other documented designs.
 - Write reviews of tools, software, video games, etc.

• St. Paul's



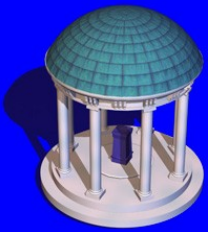
The Great Designer

- “If you want to see his monument, look around.”
- **Some handiwork!** SOD, hemoglobin, human visual system, earth as a life incubator



The Great Designer

- “If you want to see his monument, look around.”
- **Some handiwork!** SOD, hemoglobin, human visual system, earth as a life incubator
- **Good programmers don’t write sorts; they write sort generators.**



The Great Designer

- “If you want to see his monument, look around.”
- **Some handiwork!** SOD, hemoglobin, human visual system, earth as a life incubator
- **Good programmers don’t write sorts; they write sort generators.**
- **Human characteristics:** morality, failure, guilt, urge to atone, desire to worship
- **The knowledge that matters most!**

